Model Compression and Hardware Acceleration for Neural Networks: A Comprehensive Survey

Year: 2020 | Citations: 830 | Authors: By Lei Deng, Guoqi Li, Song Han, Luping Shi, Yuan Xie

---

Abstract

Domain-specific hardware is becoming a promising topic in the backdrop of improvement slow down for general-purpose processors due to the foreseeable end of Moore's Law. Machine learning, especially deep neural networks (DNNs), has become the most dazzling domain witnessing successful applications in a wide spectrum of artificial intelligence (AI) tasks. The incomparable accuracy of DNNs is achieved by paying the cost of hungry memory consumption and high computational complexity, which greatly impedes their deployment in embedded systems. Therefore, the DNN compression concept was naturally proposed and widely used for memory saving and compute acceleration. In the past few years, a tremendous number of compression techniques have sprung up to pursue a satisfactory tradeoff between processing efficiency and application accuracy. Recently, this wave has spread to the design of neural network accelerators for gaining extremely high performance. However, the amount of related works is incredibly huge and the reported approaches are quite divergent. This research chaos motivates us to provide a comprehensive survey on the recent advances toward the goal of efficient compression and execution of DNNs without significantly compromising accuracy, involving both the high-level algorithms and their applications in hardware design. In this article, we review the mainstream compression approaches such as compact model, tensor decomposition, data quantization, and network sparsification. We explain their compression principles, evaluation metrics, sensitivity analysis, and joint-way use. Then, we answer the question of how to leverage these methods in the design of neural network accelerators and present the state-of-the-art hardware architectures. In the end, we discuss several existing issues such as fair comparison, testing workloads, automatic compression, influence on security, and framework/hardware-level support, and give promising topics in this field and the possible challenges as well. This article attempts to enable readers to quickly build up a big picture of neural network compression and acceleration, clearly evaluate various methods, and confidently get started in the right way.